Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A method in a multiprocessor data processing system for asynchronous execution within a program, comprising:

executing code in a first thread in a first processor of the multiprocessor data processing system;

during the executing of the code, determining whether a first keyword exists in the code, the first keyword being a flag indicating that a subsequent code element following the first keyword may be executed out of order; and

executing the subsequent code element in a second thread in a second processor of the multiprocessor data processing system if the flag indicates that the subsequent code element may be executed out of order, such that the code and the subsequent code element are executed asynchronously with respect to one another.

2. (Currently amended) The method of claim 1, wherein the subsequent code element is one of an instruction, a block, and a method.

3. (Previously presented) The method of claim 1, wherein the first keyword is usable in both an internal definition of a method and a type definition for the method.

4. (Cancelled)

5. (Currently amended) A method in a multiprocessor data processing system for asynchronous execution within a program, comprising:

executing code in a first thread in a first processor of the multiprocessor data processing system;

determining whether a first keyword exists in the code, the first keyword indicating a code element that may be executed out of order;

executing the code element in a second thread in a second processor of the multiprocessor data processing system if the first keyword indicates the code element may be executed out of order, such that the code and the code element are executed asynchronously with respect to one another;

determining whether a second keyword exists in the code, the second keyword indicating that execution of the code element in the second thread must complete before a next code element immediately following the second keyword is executed; and

executing the next code element in the first thread after execution of the code element in the second thread completes if the second keyword indicates that execution of the code element in the second thread must complete before the next code element immediately following the second keyword is executed.

6. (Previously presented) The method of claim 5, further comprising:

determining whether a third keyword exists in the code element, the third keyword indicating a statement that may be executed out of order; and

executing the statement in a third thread.

7. (Currently amended) The method of claim 1, wherein the method is executed by ~~an~~ a run-time interpreter.

8. (Currently amended) The method of claim 7, wherein the run-time interpreter is a Java virtual machine.

9. (Original) The method of claim 1, wherein the second thread is a light weight thread.

10. (Currently amended) An apparatus for asynchronous execution within a program, comprising:

first execution means for executing, by the apparatus, code in a first thread;

determination means for determining, by the apparatus, whether a first keyword exists in the code, the first keyword being a type definition for a subsequent code element indicating that the subsequent code element following the first keyword may be executed out of order; and

second execution means for executing, by the apparatus, the subsequent code element in a second thread, such that the code and the subsequent code element are executed asynchronously with respect to one another.

11. (Currently amended) The apparatus of claim 10, wherein the subsequent code element is ~~one of an instruction, a block, and~~ a method.

12. (Previously presented) The apparatus of claim 10, wherein the first keyword is usable in both an internal definition of a method and a type definition for the method.

13. (Original) The apparatus of claim 10, wherein the first thread is executed on a first processor and the second thread is executed on a second processor.

14. (Currently amended) An apparatus for asynchronous execution within a program, comprising:

first execution means for executing, by the apparatus, code in a first thread;

determination means for determining, by the apparatus, whether a first keyword exists in the code, the first keyword indicating a code element that may be executed out of order;

second execution means for executing, by the apparatus, the code element in a second thread, such that the code and the code element are executed asynchronously with respect to one another;

means for determining, by the apparatus, whether a second keyword exists in the code, the second keyword indicating that execution of the code element in the second thread must complete before a next code element immediately following the second keyword is executed; and

means for executing, by the apparatus, the next code element in the first thread after execution of the code element in the second thread completes.

15. (Previously presented) The apparatus of claim 14, further comprising:

means for determining, by the apparatus, whether a third keyword exists in the code element, the third keyword indicating a statement that may be executed out of order; and

means for executing, by the apparatus, the statement in a third thread.

16. (Original) The apparatus of claim 10, wherein the second thread is a light weight thread.

17. (Currently amended) An apparatus for asynchronous execution within a program, comprising:

an interpreter; and

~~a program,~~ the program including a first keyword indicating a code element that may be executed out of order,

wherein the interpreter, upon detecting the first keyword, creates a light weight thread and executes the code element in the light weight thread, such that the code element executes asynchronously with respect to at least one other code element within the program.

18. (Original) The apparatus of claim 17, wherein the interpreter is a Java virtual machine.

19. (Currently amended) A computer program product, tangibly embodied in a tangible computer readable medium, for asynchronous execution within a program, comprising:

instructions for executing code in a first thread;

instructions for determining, during the executing of the code, whether a first keyword exists in the code, the first keyword being a flag indicating that a subsequent code element following the first keyword may be executed out of order; and

instructions for executing the subsequent code element in a second thread, such that the code and the subsequent code element are executed asynchronously with respect to one another.

20. (Original) The computer program product of claim 19, wherein the first thread is executed on a first processor and the second thread is executed on a second processor.

21. (Currently amended) A computer program product, tangibly embodied in a tangible computer readable medium, for asynchronous execution within a program, comprising:

instructions for executing code in a first thread;

instructions for determining whether a first keyword exists in the code, the first keyword indicating a code element that may be executed out of order;

instructions for executing the code element in a second thread, such that the code and the code element are executed asynchronously with respect to one another;

instructions for determining whether a second keyword exists in the code, the second keyword indicating that execution of the code element in the second thread must complete before a next code element immediately following the second keyword is executed; and

instructions for executing the next code element in the first thread after execution of the code element in the second thread completes.

22. (Previously presented) The computer program product of claim 21, further comprising:

instructions for determining whether a third keyword exists in the code element, the third keyword indicating a statement that may be executed out of order; and

instructions for executing the statement in a third thread.